Remarks

This Amendment responds to the final Office Action ("the Action") mailed February 26, 2007. Reconsideration of the application is respectfully requested in view of the following remarks. Claims 1-52 are pending in the application. No claims have been allowed. Claims 1, 17, 29, and 44 are independent.

Cited Art

U.S. Patent App. Pub No. 2002/0133639 to Breslau et al. ("Breslau") is entitled "Method and System for Migrating an Object Between a Split Status and a Merged Status."

Amendments

Editorial amendments have been made to the claims to address the rejection under 35 U.S.C. § 101 and for claim language consistency. No amendments have been made in response to the rejections under 35 U.S.C. § 102(e). No new matter is added.

Statutory Matter Rejection

Claims 1-43 have been rejected by the Action under 35 U.S.C. § 101 as being directed to non-statutory subject matter. Specifically, the Action alleges that there exists a question as to whether claims 1, 17, and 29 are "directed merely to an abstract idea that is not tied to an environment or machine which would result in a practical application that would produce a useful, concrete, and tangible result." [Action, at §6, page 2.]

Applicants respectfully disagree with the Action that the claims do not produce a useful, concrete, and tangible result. For instance, Applicants believe that the "coallocation solution" recited in the claims is sufficient to satisfy the requirements of 35 U.S.C. § 101. However, in the interest of expediting prosecution, Applicants have amended independent claims 1, 17, and 29 to recite, among other things:

> enforcing the coallocation solution during subsequent execution of the computer program by coallocating sets of heap objects in heap memory arenas in order to improve locality for accesses of the heap objects.

With these amendments, the claims recite language which produces a useful, concrete, and tangible result, namely the allocation of objects into specific memory arenas according to the

produced coallocation solution. Therefore, Applications believe claims 1, 17, and 29 are allowable under 35 U.S.C. § 101. Additionally, claims 2-16, 18-28, and 30-43, each of which depend from these independent claims but recite additional language, are thus also allowable under 35 U.S.C. § 101.

For at least these reasons, Applicants respectfully request that the rejection of claims 1-43 under 35 U.S.C. § 101 be withdrawn and that the claims be allowed.

Rejection of Claims Under 35 U.S.C. § 102 over Breslau

The Action rejects claims 1-52 under 35 U.S.C. § 102(e) as being anticipated by Breslau. Applicants respectfully disagree and traverse the rejection. For a 102(b) rejection to be proper, the cited art must show each and every element as set forth in a claim. (*See* MPEP § 2131.01.) However, the cited art does not describe each and every element. Accordingly, Applicants request that the rejection be withdrawn. Claims 1, 17, 29, and 44 are independent.

Claim 1

Claim 1 recites, in part:

> receiving a temporal data reference profile for a computer program;
> detecting one or more hot data streams in the temporal data reference profile;
> *analyzing the one or more hot data streams and the temporal data reference profile to determine a coallocation solution for allocations in heap memory*; . . . .

[Emphasis added.] For example, the Application describes examples of determination of coallocation solutions at page 31:

> The tool 1100 includes one or more analysis modules 1120 for analyzing the profile 1115 and determining one or more cache-conscious coallocation solutions 1125. The analysis module(s) 1120 receive a profile 1115 such as a data reference trace for subsequent analysis. For example, the tool cmanal efficiently analyzes a context-free grammar representation for a profile to find hot data streams with their associated normalized heat value. The tool cmanal applies a coallocation algorithm (as described in the previous sections) to produce a collection of coallocation sets. The allocation context abstraction level (i.e., just the allocation site, or a calling context of some length l) is a tunable parameter, but may also be pre-defined. Alternatively, the analysis module(s) include other and/or additional modules.

[Application, at page 31, lines 1-10.] Particular examples of a process for determining a coallocation solution are described in the Application at Figure 4 and pages 21-23. In the examples described therein, a process is described including "comput[ing] coallocation sets and miss reduction weights," "comput[ing] partition of the set of allocation sties," "comput[ing] cache miss reduction for other layouts," and "relat[ing] cache miss reduction numbers for coallocation and other layouts." [*See*, Application, at page 21, line 18 to page 23, line 25.]

Breslau, which is directed toward the splitting and merging of objects to address load planning issues, does not teach or suggest "analyzing the one or more hot data streams and the temporal data reference profile to determine a coallocation solution for allocations in heap memory" as it does not discuss any allocation of objects. Breslau is directed to "a method and system for migrating an object between a merged status having a single instance and a split status having multiple instances." [Breslau, at page 1, paragraph 0005.] Breslau's self-described motivation is to solve problems such as load planning. [*See*, Breslau, at page 1, paragraph 0011.] Breslau also describes the need for load planning:

> Objects are shared resources that may be invoked (i.e., by invoking their methods) by other objects throughout the object-oriented computer system. The load on an object (and the corresponding load on the execution environment it is instantiated within) will therefore vary with the periodicity of invocations of the object and complexity of the methods used within the object. *Careful system planning is required such that enough instances of any particular object are available to handle the presented load. . . .*

[Breslau, at page 1, paragraph 0008; emphasis added.] Thus, Breslau is directed to solving a problem of needing *enough instances* of objects to handle program loads. To solve this problem, Breslau "comprises a method of managing the object at runtime and includes identifying a request to migrate the object between a split status and a merged status." [Breslau, at page 1, paragraph 0012.] Breslau does not discuss the usage of object splitting or merging to address cache performance, nor does it ever discuss caching.

In contrast to Breslau's stated goals, Applicants note that Breslau does not discuss "analyzing the one or more hot data streams and the temporal data reference profile to determine a coallocation solution for allocations in heap memory" as recited in claim 18. The Action, in its rejection of claim 1, cites to paragraphs 0032-0035 of page 3. However, this passage describes only the "splitting" of an object into multiple objects in order to increase processing capacity. [*See*, Breslau, at page 3, paragraph 0032.] The passage does not describe object allocation. In

fact, when Breslau later describes the instantiation of objects, it makes clear that its methods are unconcerned with the particulars of allocation:

> A next step in the process includes instantiating multiple instances of the object being split into multiple execution environments (STEP 67). For example, in FIG. 3, multiple instances of "object B" (e.g., objects B1 61, B2 63 and B3 65) are loaded into the workstations. *The creation of an object within an execution environment is conventional.*

[Breslau, at page 4, paragraph 0039.] Thus, Breslau indicates that it does not describe particular allocation of objects, let alone "a coallocation solution." In the other passage cited by the Action for the rejection of this language, paragraphs 0053-0055 of page 5, Breslau describes management of already-split objects, such as decisions about invocation of objects and informing objects of split or merged status. [*See*, Breslau, at page 5, paragraph 0053-0055.] As above, this passage does not describe object allocation, let alone a "coallocation solution." Applicants do not find further disclosure of "a coallocation solution" in Breslau. Indeed, Breslau does not once discuss the term "allocation" anywhere in its disclosure.

Applicants note that the Application does discuss "object splitting" at points in the specification. However, Applicants also note that this should not be conflated with the object splitting and merging of Breslau to the point of assuming that it teaches or suggests the recited language of claim 1. Object splitting is described, in the Application, merely as a tool used in the process of determining a coallocation solution, and does not in fact, by itself, teach or suggest "determine[ing] a coallocation solution":

> The techniques and tools extend to coallocation at object field granularity. The resulting field coallocation solution generalizes common data restructuring techniques, such as field reordering, object splitting, and object merging.

[Application, at page 10, lines 19-21.] Another example of how object splitting alone does not teach or suggest "determine[ing] a coallocation solution" can be found at page 26. Here, the Application does say that "Fields in a hot data stream are 'object split coallocatable' if object splitting (either with or without field reordering) suffices to ensure coallocation of these fields." [Application, at page 26, lines 27-28.] However, the Application goes on to qualify this statement:

> For example, in FIG. 7C, the fields $a_{3,3}$ and $a_{1,6}$ are object split coallocatable, since *splitting the fields of A, while placing the first and the third fields of instances of A contiguously, ensures coallocation* (without intervening

> instances of objects for the second field of A). Practical implementations of object splitting are described below.

[Application, at page 26, line 28 to page 27, line 2; emphasis added.] Thus, while object splitting is used to effect coallocation, the actual split fields are also explicitly allocated in such a way as to effect a coallocation solution, demonstrating that object splitting alone is insufficient to teach or suggest "determine[ing] a coallocation solution" as recited in claim 1.

For at least these reasons, Breslau does not teach or suggest "analyzing the one or more hot data streams and the temporal data reference profile to determine a coallocation solution for allocations in heap memory" and thus, Breslau does not describe each and every element as set forth in claim 1. The rejection of claim 1 over Breslau is thus improper. Applicants respectfully note that claim 1 is allowable and request that claim 1, as well as dependent claims 2-16, be allowed.

Claims 17

Claim 17 recites, in part:

> determining a coallocation solution based at least in part upon the profile, wherein the coallocation solution increases locality of object fields in a layout in memory to improve cache performance . . . .

The Action rejects claim 17 over similar passages of Breslau as in its rejection of claim 1. Thus, for at least the reasons cited above with respect to claim 1, Windows does not describe each and every element of claim 17. The rejection of claim 17 over Breslau is thus improper. Applicants respectfully note that claim 17 is allowable and request that claim 17, as well as dependent claims 18-28, be allowed.

Claims 29

Claim 29 recites, in part:

> analyzing the one or more data access patterns and the temporal data access profile to determine a coallocation solution for allocations in memory . . . .

The Action rejects claim 29 over similar passages of Breslau as in its rejection of claim 1. Thus, for at least the reasons cited above with respect to claim 1, Windows does not describe each and every element of claim 29. The rejection of claim 29 over Breslau is thus improper.

Applicants respectfully note that claim 29 is allowable and request that claim 29, as well as dependent claims 30-43, be allowed.

Claims 44

Claim 44 recites, in part:

> an analysis module for determining a coallocation solution based at least in part upon a temporal data access profile of a computer program . . . .

The Action rejects claim 44 over paragraph 0015 of Breslau, which describes "invocation persistent values" and migrating states between merged and split objects [*See*, Breslau, at page 1, paragraph 0015.] This passage does not discuss allocation, or a coallocation solution. Thus, for at least the reasons cited above with respect to claim 1, Windows does not describe each and every element of claim 44. The rejection of claim 44 over Breslau is thus improper. Applicants respectfully note that claim 44 is allowable and request that claim 44, as well as dependent claims 45-52, be allowed.

Request for Interview

If any issues remain, the Examiner is formally requested to contact the undersigned attorney prior to issuance of the next Office Action in order to arrange a telephonic interview. It is believed that a brief discussion of the merits of the present application may expedite prosecution. Applicants submit the foregoing formal Amendment so that the Examiner may fully evaluate Applicants' position, thereby enabling the interview to be more focused.

This request is being submitted under MPEP § 713.01, which indicates that an interview may be arranged in advance by a written request.

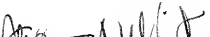
Conclusion

Claims 1-52 are allowable. Applicants respectfully request allowance of the application.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By 
Stephen A. Wight
Registration No. 37,759